

[illegible]

[illegible]

```

LL          IIIII
LL          IIIII
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LLLLLLLLLLL IIIII
LLLLLLLLLLL IIIII
SSSSSSSSS
SSSSSSSSS
SS
SS
SS
SS
SSSSSS
SSSSSS
SS
SS
SS
SS
SSSSSSSSS
SSSSSSSSS

```



```
1 0001 0 MODULE STR$COMPARE_CASE_BLIND ( ! Compare Case-blind
2 0002 0
3 0003 0 IDENT = '1-003' ! File: STR$COMCAS.B32 Edit: DG1003
4 0004 0
5 0005 0 ) =
6 0006 1 BEGIN
7 0007 1
8 0008 1
9 0009 1 *****
10 0010 1 *
11 0011 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
12 0012 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
13 0013 1 * ALL RIGHTS RESERVED.
14 0014 1 *
15 0015 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
16 0016 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
17 0017 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
18 0018 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
19 0019 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
20 0020 1 * TRANSFERRED.
21 0021 1 *
22 0022 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
23 0023 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
24 0024 1 * CORPORATION.
25 0025 1 *
26 0026 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
27 0027 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
28 0028 1 *
29 0029 1 *
30 0030 1 *****
31 0031 1
32 0032 1
33 0033 1 ++
34 0034 1 FACILITY: String support library
35 0035 1
36 0036 1 ABSTRACT: This module contains a routine to perform case-blind
37 0037 1 comparisons of 2 input strings of any supported class and dtype.
38 0038 1
39 0039 1 ENVIRONMENT: User mode, AST reentrant.
40 0040 1
41 0041 1 AUTHOR: R. Reichert, CREATION DATE: 11-Aug-1982
42 0042 1
43 0043 1 MODIFIED BY:
44 0044 1
45 0045 1 1-001 - Original. RKR 11-Aug-1982.
46 0046 1 1-002 - Remove STR$CASE BLIND COMPARE EQL. Improvements to code
47 0047 1 in STR$CASE BLIND COMPARE. RKR 24-Nov-1982.
48 0048 1 1-003 - Enhance to support DEC Multinational Char. Set. DG 1-Nov-1983.
49 0049 1 --
50 0050 1 !<BLF/PAGE>
```



```
52 0051 1 |
53 0052 1 | SWITCHES:
54 0053 1 |
55 0054 1 |
56 0055 1 SWITCHES ADDRESSING MODE
57 0056 1 (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
58 0057 1 |
59 0058 1 |
60 0059 1 | LINKAGES
61 0060 1 |
62 0061 1 REQUIRE 'RTLIN:STRLNK'; ! JSB linkages
63 0246 1 |
64 0247 1 |
65 0248 1 | TABLE OF CONTENTS:
66 0249 1 |
67 0250 1 |
68 0251 1 FORWARD ROUTINE
69 0252 1 STR$CASE_BLIND_COMPARE; ! With blank fill for unequal lengths
70 0253 1 |
71 0254 1 |
72 0255 1 | INCLUDE FILES:
73 0256 1 |
74 0257 1 |
75 0258 1 REQUIRE 'RTLIN:RTLPSECT'; ! Declare PSECTS code
76 0353 1 REQUIRE 'RTLIN:STRMACROS'; ! String macros
77 1269 1 LIBRARY 'RTLSTARLE'; ! STARLET library for macros and symbols
78 1270 1 |
79 1271 1 |
80 1272 1 | MACROS
81 1273 1 |
82 M 1274 1 MACRO UPCASE (IN_BYTE) =
83 M 1275 1 BEGIN
84 M 1276 1 IF (CH$RCHAR(.IN_BYTE) GEQU %C'a' AND
85 M 1277 1 CH$RCHAR(.IN_BYTE) LEQU %C'z') OR
86 M 1278 1 (CH$RCHAR(.IN_BYTE) GEQU %X'EO' AND
87 M 1279 1 CH$RCHAR(.IN_BYTE) LEQU %X'FD' AND
88 M 1280 1 CH$RCHAR(.IN_BYTE) NEQ %X'FO')
89 M 1281 1 THEN CH$RCHAR_A(IN_BYTE) - (%C'a' - %C'A')
90 M 1282 1 ELSE CH$RCHAR_A(IN_BYTE)
91 M 1283 1 END % ;
92 1284 1 |
93 1285 1 |
94 1286 1 |
95 1287 1 | EQUATED SYMBOLS: NONE
96 1288 1 |
97 1289 1 |
98 1290 1 |
99 1291 1 | PSECT DECLARATIONS
100 1292 1 |
101 1293 1 |
102 1294 1 DECLARE_PSECTS (STR);
103 1295 1 |
104 1296 1 |
105 1297 1 | OWN STORAGE: NONE
106 1298 1 |
107 1299 1 |
108 1300 1 |
```

STR\$COMPARE_CAS
1-003

: 109

1301 1 ! EXTERNAL ROUTINE : NONE

D 1
16-Sep-1984 01:31:40
14-Sep-1984 12:40:01

VAX-11 Bliss-32 V4.0-742
[LIBRTL.SRC]STRCOMCAS.B32;1

Page 3
(2)


```
111 1302 1 GLOBAL ROUTINE STR$CASE_BLIND_COMPARE ( ! Case-blind compare with
112 1303 1 ! blank fill
113 1304 1
114 1305 1 STRING1, ! pointer to 1st string descriptor
115 1306 1 STRING2 ! pointer to 2nd string descriptor
116 1307 1
117 1308 1 ) : =
118 1309 1
119 1310 1 ++
120 1311 1 FUNCTIONAL DESCRIPTION:
121 1312 1 This routine compares two upcased source strings of any
122 1313 1 supported data type and class, and returns a
123 1314 1
124 1315 1 -1 if string1 < string2
125 1316 1 0 if both are the same with blank fill for shorter
126 1317 1 1 if string1 > string2.
127 1318 1
128 1319 1 CALLING SEQUENCE:
129 1320 1
130 1321 1 MATCH.wl.v = STR$CASE_BLIND_COMPARE ( STRING1.rt.dx,
131 1322 1 STRING2.rt.dx)
132 1323 1
133 1324 1 FORMAL PARAMETERS:
134 1325 1
135 1326 1 STRING1.rt.dx pointer to 1st string descriptor
136 1327 1 STRING2.rt.dx pointer to 2nd string descriptor
137 1328 1
138 1329 1 IMPLICIT INPUTS:
139 1330 1
140 1331 1 NONE
141 1332 1
142 1333 1 IMPLICIT OUTPUTS:
143 1334 1
144 1335 1 NONE
145 1336 1
146 1337 1 ROUTINE VALUE:
147 1338 1
148 1339 1 MATCH.wl.v -1 if string1 < string2
149 1340 1 0 if both are the same with blank fill
150 1341 1 1 if string1 > string2
151 1342 1
152 1343 1 SIDE EFFECTS:
153 1344 1
154 1345 1 May signal STR$_ILLSTRCLA on bad string class
155 1346 1 --
156 1347 2 BEGIN
157 1348 2 MAP
158 1349 2 STRING1: REF BLOCK [ ,BYTE],
159 1350 2 STRING2: REF BLOCK [ ,BYTE];
160 1351 2
161 1352 2 LOCAL
162 1353 2 STRING1_LEN : WORD, ! Length of STRING1
163 1354 2 STRING1_ADDR, ! Address of STRING1
164 1355 2 STRING2_LEN : WORD, ! Length of STRING2
165 1356 2 STRING2_ADDR, ! Address of STRING2
166 1357 2 STATUS, ! Status to be returned
167 1358 2 SHORTER ; ! shorter of STRING1_LEN and STRING2_LEN
```

```
168 1359 2
169 1360 2
170 1361 2 + Extract length and addresses of strings, and find length of shorter
171 1362 2 -
172 1363 2 $STR$GET_LEN_ADDR ( STRING1, STRING1_LEN, STRING1_ADDR ) ;
173 1364 2 $STR$GET_LEN_ADDR ( STRING2, STRING2_LEN, STRING2_ADDR ) ;
174 1365 2
175 1366 2 SHORTER = MINU ( .STRING1_LEN, .STRING2_LEN ) ;
176 1367 2
177 1368 2 +
178 1369 2 Compare two strings character by character, upcasing each character
179 1370 2 while inspecting it.
180 1371 2 -
181 1372 2 INCR I FROM 1 TO .SHORTER
182 1373 2 DO
183 1374 2 BEGIN ! loop
184 1375 2 LOCAL
185 1376 2 CHAR1 : BYTE,
186 1377 2 CHAR2 : BYTE ;
187 1378 2 CHAR1 = UPCASE ( STRING1_ADDR ) ;
188 1379 2 CHAR2 = UPCASE ( STRING2_ADDR ) ;
189 1380 2 IF .CHAR1 LSSU .CHAR2
190 1381 2 THEN RETURN (-1)
191 1382 2 ELSE IF .CHAR1 GTRU .CHAR2
192 1383 2 THEN RETURN (1);
193 1384 2 END; ! loop
194 1385 2
195 1386 2 +
196 1387 2 If we fall out of loop, strings matched all the way across to the
197 1388 2 length of the shorter. If their lengths are equal, they're equal.
198 1389 2 IF .STRING1_LEN EQLU .STRING2_LEN
199 1390 2 THEN RETURN 0 ; ! length and contents match
200 1391 2
201 1392 2 +
202 1393 2 If their lengths are not equal, and what remains of the longer is
203 1394 2 blanks, then they are equal, else shorter is less.
204 1395 2 -
205 1396 2
206 1397 2 IF .STRING1_LEN LSSU .STRING2_LEN
207 1398 2 THEN
208 1399 2 BEGIN ! STRING1 is shorter
209 1400 2 IF CH$EQL ( 0, .STRING2_ADDR,
210 1401 2 .STRING2_LEN - .SHORTER, .STRING2_ADDR,
211 1402 2 STR$K_FILL_CHAR )
212 1403 2 THEN RETURN 0 ! STRING1 (with blank fill) = STRING2
213 1404 2 ELSE RETURN -1 ; ! STRING1 < STRING2
214 1405 2 END ! STRING1 is shorter
215 1406 2 ELSE
216 1407 2 BEGIN ! STRING2 is shorter
217 1408 2 IF CH$EQL ( 0, .STRING1_ADDR,
218 1409 2 .STRING1_LEN - .SHORTER, .STRING1_ADDR,
219 1410 2 STR$K_FILL_CHAR )
220 1411 2 THEN RETURN 0 ! STRING1 = STRING2 (with blank fill)
221 1412 2 ELSE RETURN 1 ; ! STRING1 > STRING2
222 1413 2 END ; ! STRING2 is shorter
223 1414 2
224 1415 2 END ; ! STR$CASE_BLIND_COMPARE
```


				.TITLE	STR\$COMPARE_CASE_BLIND	
				.IDENT	\1-003\	
				.EXTRN	STR\$ANALYZE_SDESC_R1	
				.PSECT	_STR\$CODE,NOWRT, SHR, PIC,2	
03FC 00000				.ENTRY	STR\$CASE_BLIND_COMPARE, Save R2,R3,R4,R5,-	1302
59	00000000G	00	9E 00002	MOVAB	R6,R7,R8,R9	:
50	04	AC	D0 00009	MOVL	STR\$ANALYZE_SDESC_R1, R9	1363
02	03	A0	91 0000D	CMPB	STRING1, R0	:
		09	1A 00011	BGTRU	3(R0), #2	:
57		60	B0 00013	BRB	1\$	:
52	04	A0	D0 00016	MOVW	(R0), STRING1_LEN	:
		08	11 0001A	MOVL	4(R0), STRING1_ADDR	:
		69	16 0001C	BRB	2\$	:
52		51	D0 0001E	JSB	STR\$ANALYZE_SDESC_R1	:
57		50	D0 00021	MOVL	R1, R2	:
50	08	AC	D0 00024	MOVL	R0, STRING1_LEN	:
02	03	A0	91 00028	MOVL	STRING2, R0	1364
		09	1A 0002C	CMPB	3(R0), #2	:
54		60	B0 0002E	BGTRU	3\$	:
51	04	A0	D0 00031	MOVW	(R0), STRING2_LEN	:
		05	11 00035	MOVL	4(R0), STRING2_ADDR	:
		69	16 00037	BRB	4\$	:
54		50	D0 00039	JSB	STR\$ANALYZE_SDESC_R1	:
50		57	3C 0003C	MOVL	R0, STRING2_LEN	:
50		54	B1 0003F	MOVZWL	STRING1_LEN, R0	1366
		03	1E 00042	CMPW	STRING2_LEN, R0	:
50		54	3C 00044	BGEQU	5\$	:
58		50	D0 00047	MOVZWL	STRING2_LEN, R0	:
		56	D4 0004A	MOVL	R0, SHORTER	:
		5F	11 0004C	CLRL	I	1372
61	8F	62	91 0004E	BRB	15\$	:
		06	1F 00052	CMPB	(STRING1_ADDR), #97	1378
7A	8F	62	91 00054	BLSSU	7\$	:
		12	1B 00058	CMPB	(STRING1_ADDR), #122	:
E0	8F	62	91 0005A	BLEQU	8\$	:
		14	1F 0005E	CMPB	(STRING1_ADDR), #224	:
FD	8F	62	91 00060	BLSSU	9\$	:
		0E	1A 00064	CMPB	(STRING1_ADDR), #253	:
F0	8F	62	91 00066	BGTRU	9\$	:
		08	13 0006A	CMPB	(STRING1_ADDR), #240	:
		82	9A 0006C	BEQL	9\$	:
50		20	C2 0006F	MOVZBL	(STRING1_ADDR)+, R0	:
50		03	11 00072	SUBL2	#32, R0	:
		82	9A 00074	BRB	10\$	:
55		50	90 00077	MOVZBL	(STRING1_ADDR)+, R0	:
61	8F	61	91 0007A	MOVB	R0, CHART	:
		06	1F 0007E	CMPB	(STRING2_ADDR), #97	1379
7A	8F	61	91 00080	BLSSU	11\$	:
		12	1B 00084	CMPB	(STRING2_ADDR), #122	:
E0	8F	61	91 00086	BLEQU	12\$	:
		14	1F 0008A	CMPB	(STRING2_ADDR), #224	:
				BLSSU	13\$	:

STR\$COMPARE_CAS
1-003

H 1
16-Sep-1984 01:31:40 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:40:01 [LIBRTL.SRC]STR\$COMCAS.B32;1

Page 7
(3)

FD	8F	61	91	0008C	CMPB	(STRING2_ADDR), #253	:
		0E	1A	00090	BGTRU	13\$	:
FO	8F	61	91	00092	CMPB	(STRING2_ADDR), #240	:
		08	13	00096	BEQL	13\$	:
	50	81	9A	00098	12\$: MOVZBL	(STRING2_ADDR)+, R0	:
	50	20	C2	0009B	SUBL2	#32, R0	:
		03	11	0009E	BRB	14\$	:
	50	81	9A	000A0	13\$: MOVZBL	(STRING2_ADDR)+, R0	:
	53	50	90	000A3	14\$: MOVB	R0, CHAR2	:
	53	55	91	000A6	CMPB	CHAR1, CHAR2	: 1380
		1B	1F	000A9	BLSSU	16\$	:
		2B	1A	000AB	BGTRU	18\$	: 1382
9D	56	58	F3	000AD	15\$: AOBLEQ	SHORTER, I, 6\$	: 1372
	54	57	B1	000B1	CMPW	STRING1_LEN, STRING2_LEN	: 1389
		26	13	000B4	BEQL	19\$	:
		12	1E	000B6	BGEQU	17\$	: 1397
	50	54	3C	000B8	MOVZWL	STRING2_LEN, R0	: 1401
	50	58	C2	000BB	SUBL2	SHORTER, R0	:
50	20	61	00	2D	000BE	CMPC5	#0, (STRING2_ADDR), #32, R0, (STRING2_ADDR) : 1400
		61		000C3			:
		16	13	000C4	BEQL	19\$	:
	50	01	CE	000C6	16\$: MNEGL	#1, R0	: 1404
			04	000C9	RET		: 1407
	50	57	3C	000CA	17\$: MOVZWL	STRING1_LEN, R0	: 1409
	50	58	C2	000CD	SUBL2	SHORTER, R0	:
50	20	62	00	2D	000D0	CMPC5	#0, (STRING1_ADDR), #32, R0, (STRING1_ADDR) : 1408
		62		000D5			:
		04	13	000D6	BEQL	19\$	:
	50	01	D0	000D8	18\$: MOVL	#1, R0	: 1412
			04	000DB	RET		: 1407
		50	D4	000DC	19\$: CLRL	R0	: 1415
			04	000DE	RET		:

; Routine Size: 223 bytes, Routine Base: _STR\$CODE + 0000

: 226 1416 1 END
: 227 1417 0 ELUDOM

! of module STR\$COMPARE_CASE_BLIND

PSECT SUMMARY

Name	Bytes	Attributes
_STR\$CODE	223	NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	4	0	581	00:00.7

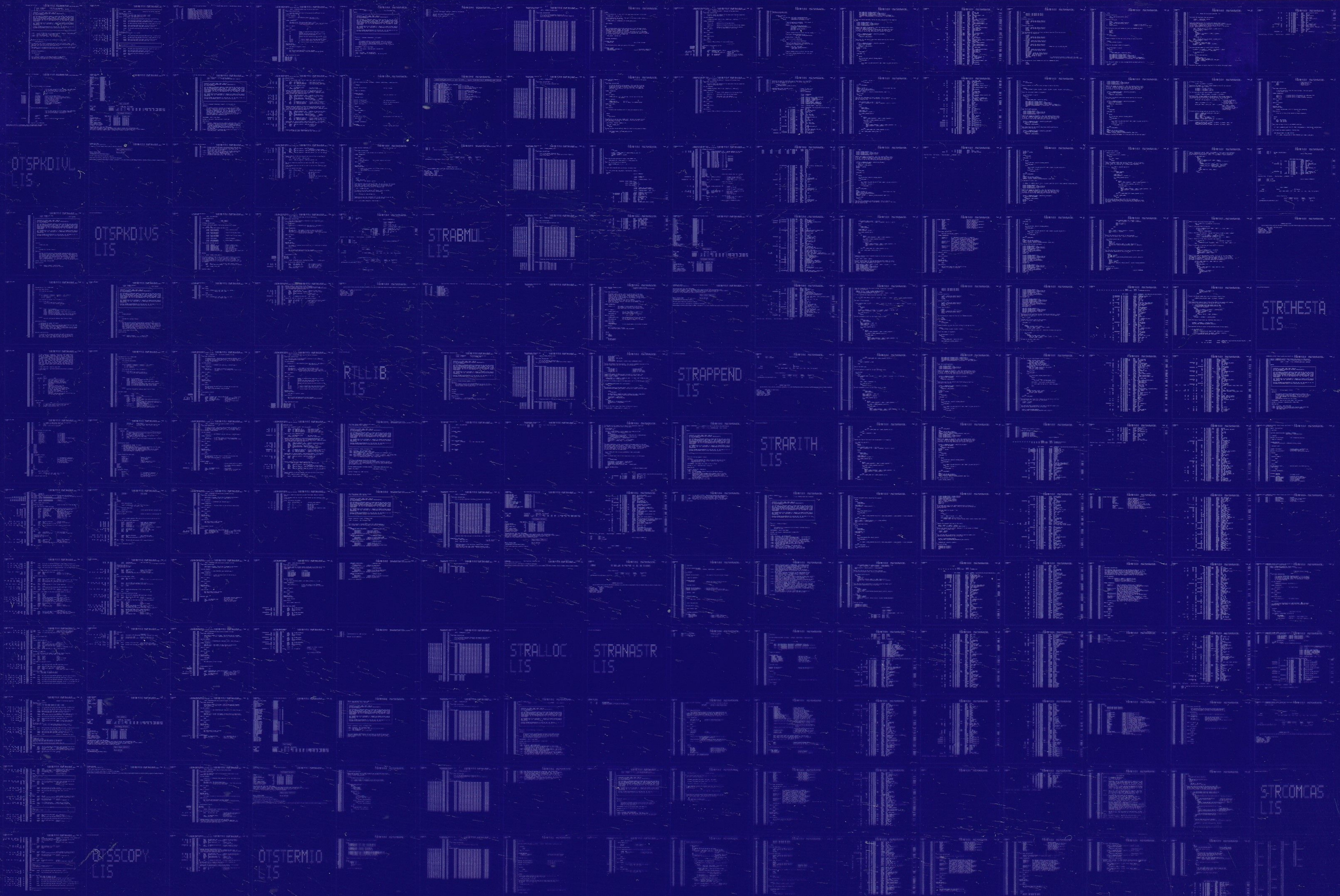
COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LISS:STRCOMCAS/OBJ=OBJ\$:STRCOMCAS MSRC\$:STRCOMCAS/UPDATE=(ENH\$:STRCOMCAS)

: Size: 223 code + 0 data bytes
: Run Time: 00:06.3
: Elapsed Time: 00:28.9
: Lines/CPU Min: 13473
: Lexemes/CPU-Min: 35220
: Memory Used: 118 pages
: Compilation Complete

0213 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



0214 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

